Immunity by Design: An Artificial Immune System

Steven A. Hofmeyr and **Stephanie Forrest**
Dept. of Computer Science
University of New Mexico
Albuquerque, NM 87131-1386
{steveah,forrest}@cs.unm.edu

Abstract

We describe an artificial immune system (AIS) that is distributed, robust, dynamic, diverse and adaptive. It captures many features of the vertebrate immune system and places them in the context of the problem of protecting a network of computers from illegal intrusions.

1 INTRODUCTION

The immune system is highly complicated and appears to be precisely tuned to the problem of detecting and eliminating infections. We believe that it also provides a compelling example of a distributed information-processing system, one which we can study for the purpose of designing better artificial adaptive systems. A important and natural application domain for adaptive systems is that of computer security. A computer security system should protect a machine or set of machines from unauthorized intruders and foreign code, which is similar in functionality to the immune system protecting the body (*self*) from invasion by inimical microbes (*nonself*). Because of this compelling similarity, we have designed an "artificial immune system" (AIS) to protect computer networks based on immunological principles, algorithms and architecture.

In designing this system, we wish to adhere to certain principles which we have extracted from our study of immunology: The immune system is *diverse*, which greatly improves robustness, on both a population and individual level, for example, different people are vulnerable to different microbes; it is *distributed*, consisting of many components that interact locally to provide global protection, so there is no central control and hence no single point of failure; it is *error tolerant* in that a few mistakes in classification and response are not catastrophic; it is *dynamic*, i.e. individual components are continually created, destroyed, and are circulated throughout the body, which increases the temporal and spatial diversity of the immune system allowing it to discard components that are useless or dangerous and improve on existing components; it is *self-protecting*, i.e. the same mechanisms that protect the body also protect the immune system itself; and it is *adaptable*, i.e. it can learn to recognize and respond to new microbes, and retain a memory of those microbes to facilitate future responses.

We regard these principles as general guidelines for design. Sometimes we can incorporate these principles by using algorithms or mechanisms copied directly from immunology, but at other times new algorithms are required. We are not primarily concerned with mimicking the immune system in all its details; rather, we are trying to capture those aspects of the immune system that are most relevant to constructing a robust distributed adaptive system.

In earlier papers we presented results from this research program in the context of computer security (e.g., [4]), deemphasizing more general considerations. The goal of this paper is to rectify that, making the biological connections more concrete and emphasizing the adaptive systems framework. In the next section (2) we describe the organization of our AIS, in the context of a specific application; most of what is described has been implemented, but some of the ideas are still speculative. The results of testing the system out in a real environment are described in Section 3, and the paper concludes with a discussion of the AIS, including its relation to classifier systems [9].

2 ARCHITECTURE

Before outlining the architecture and algorithms of our adaptive immune system (AIS), we must first consider the environment in which the AIS will exist. To preserve generality, we represent both the protected system (self) and infectious agents (nonself) as dynamically changing sets of bit strings. In cells of the body the profile of expressed proteins (self) changes over time, and likewise, we expect our set of protected strings to vary over time. Similarly, the body is subjected to different kinds of infections over time; we can view nonself as a dynamically changing set of strings.

Although we can, in principle, completely specify our immune system architecture based on this abstract representation of self and nonself as sets of bit strings, it is perhaps helpful to have a specific example in mind—one that guides specific implementation decisions in order to make the system concrete enough to test in a real environment.

2.1 APPLICATION DOMAIN: NETWORK SECURITY

The most natural domain in which to begin applying immune system mechanisms is computer security, where the analogy between protecting the body and protecting a normally operating computer is evident. Within this domain, we have studied several problems, including computer virus detection [6], host-based intrusion detection [5], and network security [8]. In this paper we concentrate on the latter—protecting a local-area broadcast network (LAN) from network-based attacks. Broadcast LANs have the convenient property that every location (computer) sees every packet passing through the LAN.

In this domain, we define self to be the set of normal pairwise connections (at the TCP/IP level) between computers, including connections between two computers in the LAN as well as connections between one computer in the LAN and one external computer (Figure 1). A connection is defined in terms of its "data-path triple"—the source IP address, the destination IP address, and the service (or port) by which the computers communicate. In our representation, this information is compressed to a single 49-bit string which unambiguously defines the connection. Self is then the set of normally occurring connections observed over time on the LAN, each connection being represented by a 49-bit string. Similarly, nonself is also a set of connections (using the same 49-bit representation), the difference being that nonself consists of those connections, potentially an enormous number, that are not normally observed on the LAN.

2.2 MAPPING IMMUNOLOGY TO COMPUTATION

Natural immune systems consist of many different kinds of cells and molecules—lymphocytes (B-lymphocytes and T-lymphocytes), macrophages, dendritic cells, natural killer cells, mast cells, interleukins, interferons, and many others. Although these components have been identified and studied experimentally, it is not always well-understood what role they play in the overall immune response. In our AIS, we will simplify by introducing one basic type of detector cell which combines useful properties from several different immune cells. This detector cell will have several different possible states, roughly corresponding to thymocytes (immature T-lymphocytes undergoing negative selection in the thymus), naive B-lymphocytes (which have never matched foreign material), and memory B-lymphocytes (which are long-lived and easily stimulated). The natural immune system also has many different types of effector cells, which implement different immune responses (e.g., macrophage, mast-cell response, etc.), which we do not currently include in our model.

Each detector cell is represented by a single bit string of length $l = 49$ bits, and a small amount of state (see Figure 1). In effect, we are representing only the receptor region on the surface of a lymphocyte. It is this region that *binds* to foreign material, a process that we call recognition. There are many ways of implementing the detectors, for example, a detector could be a production rule, or a neural network, or an agent. We chose to implement detection (binding) as *string matching*, where each detector is a string d, and detection of a string s occurs when there is a match between s and d, according to a *matching rule*. We use string matching because it is simple and efficient to implement, and easy to analyze and understand. Obvious matching rules include Hamming distance or edit distance, but we have adopted a more immunologically plausible rule, called *r-contiguous bits* [13].

Two strings d and s match under the r-contiguous bits rule if d and s have the same symbols in at least r contiguous bit positions. The value r is a threshold and determines the specificity of the detector, which is an indication of the number of strings covered by a single detector. For example, if $r = l$, the matching is completely specific, that is, the detector will detect only a single string (itself; recall that l is the length of the detector bit string). A consequence of a partial matching rule with a threshold, such as r-contiguous bits, is that there is a trade-off between the number of detectors used, and their specificity: As the specificity of the detectors increases, so the number of detectors required to achieve a certain level of coverage also increases.

The detectors are grouped into sets, one set per machine, or host, on the LAN; each host loosely corresponds to a different location in the body[1]. Because of the broadcast assumption, each detector set is constantly exposed to the current set of connections in the LAN, which it uses as a dynamic definition of self (i.e., the observed connections in a fixed time period are analogous to the set of proteins expressed in the thymus during some period of time). Within

[1]The ability of immune system cells to circulate throughout the body is an important part of the immune system that we are currently ignoring. In our system, detectors remain in one location for their lifetime.

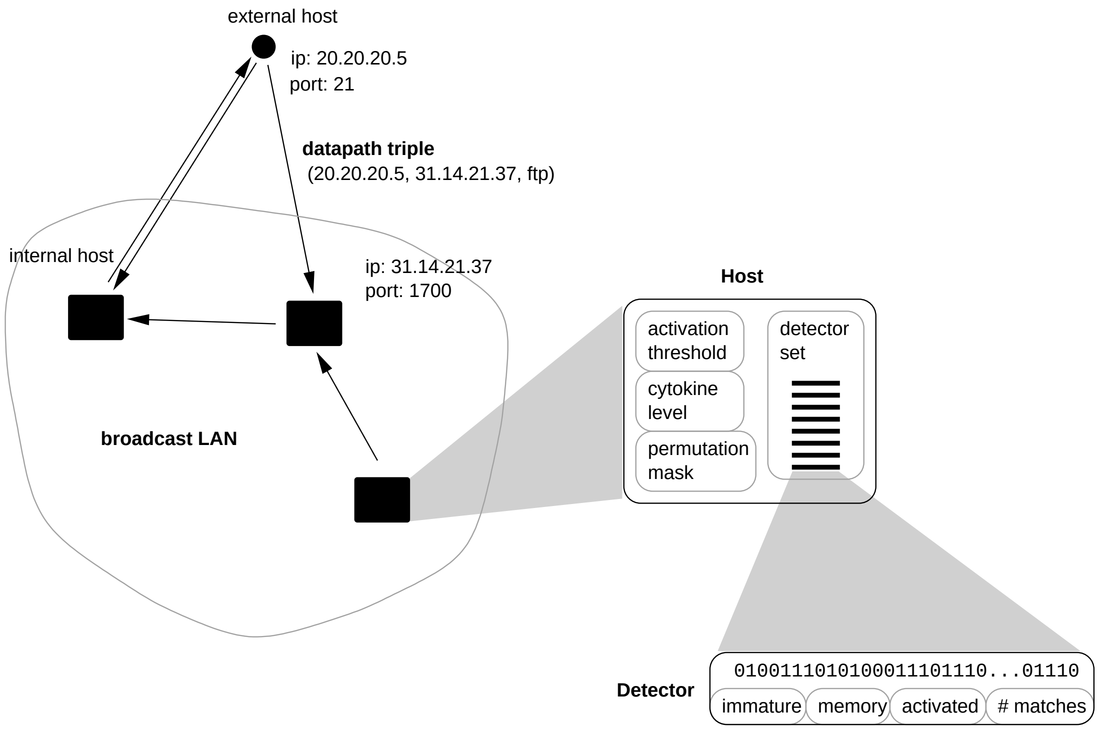

Figure 1: The Architecture of the AIS.

each detector set, new detectors, or thymocytes, are created randomly and asynchronously on a continual schedule, similar to the natural immune system. These new detectors remain *immature* for some period of time, during which they have the opportunity to match any current network connections. If a detector matches when it is immature, it is killed (deleted). This process is called *negative selection* [6], and closely resembles the negative selection of immature T-lymphocytes (thymocytes) in the thymus. A potential problem with this scheme is that a nonself packet arriving during negative selection could cause immature detectors to be erroneously eliminated. However, if we assume that nonself packets are rare (a reasonable assumption), there are likely to be other mature detectors around to detect the foreign packet. We thus have a small loss of efficiency, from needlessly deleting a valid detector, but no appreciable loss of function.

Detectors that survive this initial testing phase are promoted to mature detectors (analogous to mature T-lymphocytes leaving the thymus and mature B-lymphocytes leaving the bone marrow). Each mature detector is now a valid detector that acts independently. If a mature detector d matches a sufficient number of packets (see activation threshold below), an alarm is raised. The time for which d is a naive B-lymphocyte can be thought of as a learning phase. At the end of the learning phase, if d has failed to match a packet it is deleted, but if it has matched a sufficient number of nonself packets, it becomes a memory detector with a greatly extended lifetime. Memory detectors have a lower threshold of activation (see below), thus implementing a "secondary response" that is more sensitive and responds more aggressively than naive detectors to previously seen strings. Although these memory detectors are desirable, a large fraction of naive detectors must always be present, because the naive detectors are necessary for the detection of novel foreign packets, i.e. they are essential to anomaly detection.

2.3 INCOMPLETE SELF SETS

Both the natural immune system and our AIS face the problem of "incomplete self sets." When T lymphocytes undergo negative selection in the thymus, they are exposed to most but not all of the proteins in the body. Consequently, the negative selection process can be incomplete in the sense that a lymphocyte could survive negative selection but still be reactive against a legitimate self protein (one that was not presented in the thymus) potentially leading to an auto-immune reaction. In our AIS, such an auto-immune reaction is called a *false positive*. False positives arise if we train the system on an incomplete description of self, and then encounter new but legitimate patterns. We would like the system to be tolerant of such minor, legitimate new patterns, but still detect abnormal activity, and we have implemented two methods designed to overcome this problem: Activation thresholds and adaptive thresholds.

Activation thresholds are similar in function to avidity thresholds in lymphocytes. A lymphocyte is covered with many identical receptors, and it is only activated when sufficiently many receptors are bound to pathogens, i.e. when the avidity threshold for binding is exceeded. Analogously, each detector in the AIS must match multiple times before it is activated. Each detector records the number of times it matches, and it raises an alarm only when the number of matches exceeds the activation threshold, which is stored locally for each detector set. Once a detector has raised an alarm, it returns its match count to zero. This mechanism has a time horizon: Over time the count of matches slowly returns to zero. Thus, only repeated occurrences of structurally similar and temporally clumped strings will trigger the detection system.

However, some attacks may be launched from many different machines, in which case the first method is unlikely to be successful. To detect such distributed coordinated attacks, we introduce a second method, called *adaptive activation* (labeled *cytokine level* in Figure 1). Whenever the match count of a detector goes from 0 to 1, the local activation threshold is reduced by one. Hence, each different detector that matches for the first time "sensitizes" the detection system, so that all detectors on that machine are more easily activated in future. This mechanism also has a time horizon; over time, the activation threshold gradually returns to its default value. Thus, this method will detect diverse activity from many different sources, provided that activity happens within a certain period of time. This mechanism roughly captures the role that inflammation, cytokines, and other molecules play in increasing or decreasing the sensitivity of individual immune system lymphocytes within a physically local region.

2.4 LEARNING MECHANISMS

Negative-selection and the maturation of naive cells into memory cells are two simple learning mechanisms used by the immune system. A third form of immune-system learning, one that resembles a genetic algorithm (without crossover), is incorporated into our model—affinity maturation. In its simple form, detectors compete against one another for foreign packets, just as lymphocytes compete to bind foreign antigen. In the case where two detectors simultaneously match the same packet, the one with the closest match (greatest fitness) wins. This introduces pressure for more specific matching into the system, causing the system to discriminate more precisely between self and nonself. We propose, although we have not yet implemented this, that successful detectors (those that bind many foreign packets) will undergo proliferation (making copies and migrating to other computers) and somatic hypermutation (copying with a high mutation rate).

The concept of a *second signal*, known as *co-stimulation*, is often used to explain certain immunological responses. One example of a second signal is a T-helper lymphocyte. When a B-lymphocyte (that is possibly a mutated descendant of an earlier lymphocyte that survived negative selection) binds a foreign peptide (the first signal), it requires a T-helper lymphocyte (that has been censored against self in the thymus) in order to trigger an immune response. This second-signal system prevents mutating B-lymphocyte lines from incorrectly reacting against self. In our system, we use a human as the second signal. When a detector raises an alarm, there is some chance that it is a false alarm (auto-immune reaction). Before taking action, the AIS waits a fixed amount of time (say 24 hours) for a co-stimulatory signal, which in the current implementation is an email message from a human. If the signal is received (confirming the anomaly), the detector enters the competition to become a memory detector, but if it loses the competition, it remains naive and has its match count reset to 0. If the second signal is not received, the AIS assumes that it was a false alarm and destroys the detector (as in the natural immune system).

It might seem more natural to send messages to the AIS in the case of false alarms instead of true anomalies, so that the AIS can adjust itself appropriately by immediately deleting the auto-reactive detectors. Unfortunately, this would create a vulnerability, because a malicious adversary could send signals to the AIS, labeling true foreign packets as false alarms, thus tolerizing the AIS against certain forms of attack. The form of co-stimulation that we have used is much more difficult to subvert. Because false alarms are generally much more frequent than true anomalies, our co-stimulation method has the additional advantage action by the human operator is required in the less frequent case.

Figure 2 summarizes the lifecycle of a detector. A detector is initially randomly created, and then remains immature for a certain period of time, which is the tolerization period. If the detector matches any string a single time during tolerization, it is replaced by a new randomly generated detector string. If a detector survives immaturity, it will exist for a finite lifetime. At the end of that lifetime it is replaced by a new random detector string, unless it has exceeded its match threshold and becomes a memory detector. If the activation threshold is exceeded for a mature detector, it is activated. If an activated detector does not receive costimulation, it dies (the implicit assumption is that its activation was a false positive). However, if the activated detector receives costimulation, it enters the competition (see above)

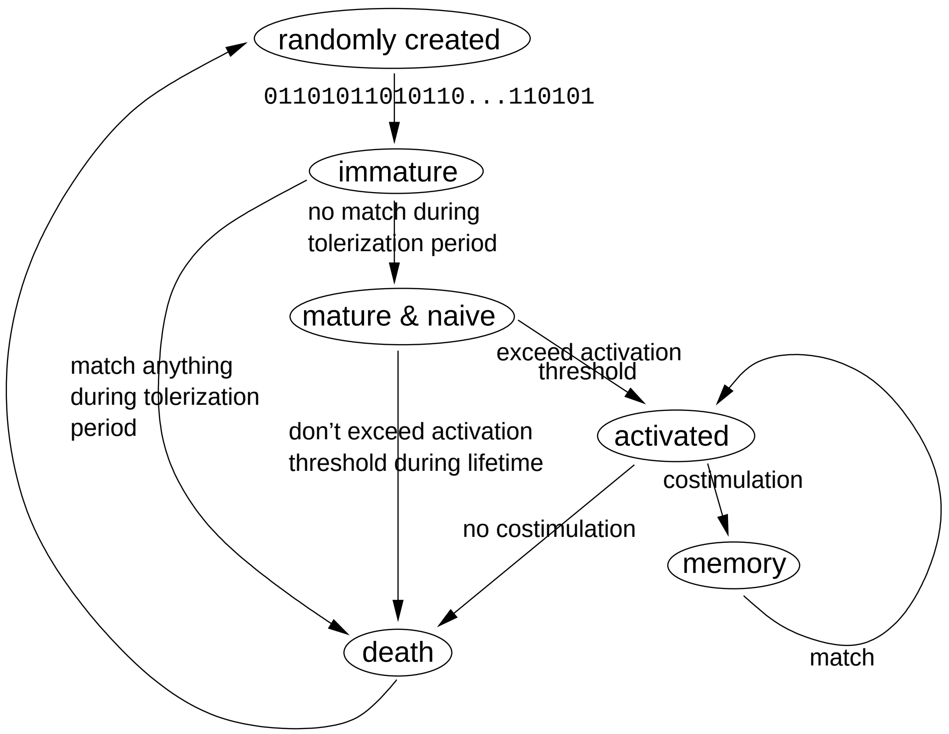

Figure 2: The Lifecycle of a Detector.

to become a memory detector with an indefinite lifespan. Memory detectors need only match once to become activated.

2.5 DISTRIBUTION AND DIVERSITY

Each of the mechanisms described above can be implemented with a single detector set running on a single location. We can trivially gain efficiency advantages by distributing the single detector set across all locations on the LAN, thus distributing the computational cost of intrusion detection. Such distribution will give linear speedup, because there are no communication costs (apart from the signaling of alarms and costimulation). However, we take advantage of another immune system feature to implement a more powerful form of distribution.

The protein *major histocompatibility complex* (MHC) plays an important role in immune systems, because it transports protein fragments (called peptides) from the interior regions of a cell to its surface, *presenting* these peptides on the cell's surface. This mechanism enables roving immune system cells to detect infections in cells without penetrating the cell membrane. There are many variations of MHC, each of which binds a slightly different class of peptides. Each individual in a population is genetically capable of making a small set of these MHC types (about ten), but the set of MHC types varies in different individuals. Consequently, individuals in a population are capable of recognizing different profiles of peptides, providing an important form of population-level *diversity*[2]. Our AIS uses permutation masks to achieve a similar kind of diversity. A permutation mask defines a permutation of the bits in the string representation of the network packets. Each detector set has a different, randomly-generated, permutation mask. One limitation of the negative-selection algorithm as originally implemented is that it can result in undetectable abnormal patterns called holes, which limit detection rates [3, 2]. Holes can exist for any symmetric, fixed-probability matching rule, but by using permutation masks, we effectively change the match rule on each host, and so overcome the hole limitation. Thus, the permutation mask controls how the network packet is presented to the detection system, which is analogous to the way different MHC types present different sets of peptides on the cell surface.

The discussion thus far has concentrated on the detection side of our AIS and ignored questions of immune response. When stimulated by lymphocytes bound to the cell surface, immune system cells secrete a wide variety of molecules

[2]For example, there are some viruses, such as the Epstein-Barr virus, that have evolved dominant peptides which cannot be bound by particular MHC types, leaving individuals who have those MHC types vulnerable to the disease [10].

known collectively as *cytokines*. These cytokines diffuse from the site where they were secreted, and in turn play a role in stimulating or suppressing other immune system cells. Thus, cells that detect pathogens can communicate using these molecular signals with cells that assist in eliminating the pathogens (e.g., mast cells, macrophages, etc.). Although we plan to extend our model in the future to include this kind of signaling and response, the current model eliminates this complication (except for the adaptive threshold).

3 RESULTS

The AIS described in Section 2 has been implemented and tested out on a subnet of the Computer Science department at the University of New Mexico, consisting of 50 machines on a switched segment. All analysis reported here was conducted off-line, although an on-line prototype has developed and tested. All results described here used a system with 100 detectors per host, with a match length of 12 (i.e. $r = 12$ in the r-contiguous bits match rule), and the 49-bit detectors described earlier.

3.1 DATA SETS

Two data sets were collected: A self set consisting of normal traffic, and a nonself set consisting of traffic generated during intrusive activity. The self set was collected over 50 days, during which a total of 2.3 million TCP connections were logged, each of which is a datapath triple. These 2.3 million datapaths were filtered down to 1.5 million datapaths. The filtering removed several classes of noisy traffic sources, such as web servers and ftp servers, because these are continually communicating with new hosts, and so have no stable definition of normal in terms of datapaths. The 1.5 million datapaths were mapped to 49-bit binary strings, using a mapping that grouped unassigned ports, to give a total of 3900 observed unique strings. The self set was divided into two parts: a training set (the first 43 days), and a test set (the last 7 days). During the test period, 137 new datapaths were logged, out of a total of 183000 datapaths. Each new triple occurred an average of 4 times. Thus the worst case false-positive rate would be 78 per day. Without threshold activation and co-stimulation, we observe 74 per day in our experiments, which is slightly less than expected because there are not enough detectors to give 100% detection. Adding a threshold of 10 reduces the false positives to 8 per day, and adding co-stimulation on top of this reduces the false positives further, to 4 per day. The human was given a day to respond, and it was assumed that in all false alarm cases the human did not respond. Given that each false alarm consists of a small set of anomalous packets, this rate is extremely good, especially when compared to state-of-the-art systems that are in current use [12].

Table 1: Detection of the 8 Incidents.

	TP BASIC	TP PERMUTATION
Average	0.28	0.43
Max. Possible	0.61	0.61
Incidents Detected	7/8	8/8

The nonself set was comprised of eight different intrusive incidents. Seven of these are faithful logs of real incidents that occurred on the network being studied, and one incident was synthetically generated to simulate an attack from many different locations. This simulated intrusion consisted of 200 random connections between internal hosts (the supposition was that the attackers had already penetrated at least one machine on the LAN). Most of the real attacks consisted of probing of one sort or another, particularly of services with recently reported vulnerabilities. At least one incident involved compromise of an internal machine. The traffic tested for each incident consisted of all datapaths from the first nonself datapath (the start of the incident), to the last nonself datapath. Thus, each incident reproduces the timing of the attack, as well as including all normal traffic that was interspersed throughout the attack.

3.2 EXPERIMENTAL RESULTS

Results averaging detection over all eight incidents are reported in table 3.2. All results use an activation threshold of 10. The first row reports the average true positive (TP) rate, the second row reports the maximum possible TP rate (the TP rate is limited because the incidents include self strings which will not be detectable), and the third row reports the number of incidents actually detected. To identify an incident, only some of the nonself strings need to be detected, so in a practical sense, the third row gives the effective true positive rate. The detection system clearly detects all 8 incidents when using permutation masks, even with an activation threshold of 10.

The effects of memory were tested out by simulation. The detection system was presented with the synthetic incident at time zero in the simulation, during which the true positive rate was 0.23 (averaged over 30 runs of the simulation). The system retained memory detectors from this incident and the simulation was continued. After simulating the network running for another 3 months, the detection system was again presented with the same synthetic incident. During the 3 months the memory detectors from the "primary response" were retained, but the other detectors were constantly dying and being reborn. Thus after three months the set of non-memory detectors had changed. Consequently, the true positive rate for the incident after 3 months was

0.76, suggesting that memory implemented in this manner is very useful for the "secondary response."

4 DISCUSSION

In the previous sections we described an architecture for an adaptive artificial system based on the immune system. It incorporates several forms of adaptation on different time scales, and it addresses an important problem of practical significance (network intrusion detection). Most of the features described in this paper have been incorporated into our software prototype that is currently running in real time on our departmental network. It routinely discovers outside attacks as well as interesting anomalies that are generated internally.

The AIS that we outlined in Section 2 resembles the architecture of a classifier system[9], although most of the details are different. Each detector d corresponds to the condition part of a classifier, where the match rule is r-contiguous bits instead of the traditional $1, 0, \sharp$ alphabet used in classifier systems. Our parameter r is a measure of the specificity of our detectors, much like the number of don't cares in a classifier condition is a measure of its generality. If we concatenate some bits to each detector which specify what the proper response is (analogous to different antibody *isotypes*), then each immune cell (detector plus response bits) corresponds directly to the condition/action rule format of classifier systems. In place of the message list we have a continuous flux of datapath triples that represent the current state of the environment. Currently, the only connections generated by our AIS (analagous to internally generated messages in a classifier system) are those resulting from alarms being sent to the human operator.

There is no direct analog of our negative-selection algorithm in classifier systems, except the learning rules (such as genetic algorithm and trigger conditions) under which new rules are generated. Bidding for messages in classifier systems is analogous to immune cells competing to bind to foreign datapaths. Likewise, we introduce pressure for specificity, which is reminiscent of classifier systems, by allowing the more specific match to win the competition.

The role of the bucket brigade (credit assignment) and the genetic algorithm is played by our affinity maturation model of learning, although ours is simpler in the sense that we assign credit directly from the environment to the detectors, and do not pass strength among immune cells. A more direct analog of the bucket brigade would occur if we tried to build up idiotypic networks of immune cell in which immune cells stimulate and repress other immune cells, as Jerne proposed [11]. Although this is appealing from an adaptive design perspective, there is little if any experimental evidence that such networks exist in natural immune systems. Our plan is to incorporate internal feedbacks and self-regulation by extending the cytokine system (we saw a primitive form of this in the adaptive threshold). Permutation masks have no direct analog in classical classifier systems. Although the mapping is not $1-1$, we believe that the AIS we have described in this paper captures many of the important properties of classifier systems and provides an interesting point of comparison.

Our starting point for this line of research was a collection of pressing unsolved problems in computer security. Over the past several years we have designed and built several successful solutions to real computer security problems. Armed with that experience, we have shown here how to embed an architecture for adaptive behavior in a real-time environment with live agents (computers and the humans who operate them). We follow Brooks and others [7, 1] in believing that it is fruitless to design intelligent systems in isolation from the environments in which they exist, and we believe that research on classifier systems has suffered from too loose a coupling with live environments. Situated intelligent artifacts are perhaps more complex to think about (because they cannot be neatly separated from their environments), but they can in some cases use their environments in ways that simplify their computations.

Moving beyond the computer network intrusion-detection application that we have described, the AIS might be applied to other classes of networks, including social networks, organizations, networks of markets, neurological networks, or ecological networks. Like our LAN with external connections, these networks consist of many components that are sparsely connected, in which there are some ordered and some random components, and in which the exact set of connections is not static. There are important computations associated with each of these networks, and they would provide an important test of the generality of our architecture in its ability to discriminate normal and abnormal activity and to respond appropriately.

Acknowledgments

The authors acknowledge the support of the Defense Advanced Research Projects Agency (grant N00014-96-1-0680), the National Science Foundation (grant IRI-9711199), the Office of Naval Research (grant N00014-99-1-0417), the IBM Partnership award, and the Intel Corporation.

References

[1] H. J. Chiel and R. D. Beer. The brain has a body: Adaptive behavior emerges from interactions of ner-

vous system, body and environment. *Trends in Neurosciences*, 20:553–557, 1997.

[2] P. D'haeseleer. An immunological approach to change detection: Theoretical results. In *Proceedings of the 9th IEEE Computer Security Foundations Workshop*, Los Alamitos, CA, 1996. IEEE Computer Society Press.

[3] P. D'haeseleer, S. Forrest, and P. Helman. An immunological approach to change detection: Algorithms, analysis and implications. In *Proceedings of the 1996 IEEE Symposium on Research in Security and Privacy*, Los Alamitos, CA, 1996. IEEE Computer Society Press.

[4] S. Forrest, S. Hofmeyr, and A. Somayaji. Computer immunology. *Communications of the ACM*, 40(10):88–96, 1997.

[5] S. Forrest, S. A. Hofmeyr, and A. Somayaji. A sense of self for unix processes. In *Proceedings of the 1996 IEEE Symposium on Research in Security and Privacy*, Los Alamitos, CA, 1996. IEEE Computer Society Press.

[6] S. Forrest, A. S. Perelson, L. Allen, and R. Cherukuri. Self-nonself discrimination in a computer. In *Proceedings of the 1994 IEEE Symposium on Research in Security and Privacy*, Los Alamos, CA, 1994. IEEE Computer Society Press.

[7] H. Hendriks-Jansen. *Catching Ourselves in the Act*. MIT Press, Cambridge, MA, 1996.

[8] Steven A. Hofmeyr. *A Immunological Model of Distributed Detection and its Application to Computer Security*. PhD thesis, Department of Computer Sciences, University of New Mexico, April 1999.

[9] J. H. Holland, K. J. Holyoak, R. E. Nisbett, and P. Thagard. *Induction: Processes of Inference, Learning, and Discovery*. MIT Press, 1986.

[10] C. A. Janeway and P. Travers. *Immunobiology: The Immune System in Health and Disease, 3rd Edition*. Current Biology Ltd., London, 1996.

[11] N. K. Jerne. Towards a network theory of the immune system. *Annals of Immunology*, 125:373–389, 1974.

[12] R. Lippman. Lincoln Laboratory intrusion detection eval- uation. http://www.ll.mit.edu/IST/ideval/index.html, 1999.

[13] J. K. Percus, O. E. Percus, and A. S. Perelson. Predicting the size of the antibody-combining region from consideration of efficient self/nonself discrimination. In *Procedings of the National Academy of Science 90*, pages 1691–1695, 1993.